AD-A132739

AMSAA

TECHNICAL REPORT NO. 373

AN EMPIRICALLY DEVELOPED FOURIER SERIES MODEL FOR DESCRIBING SOFTWARE FAILURES

LARRY H. CROW
NOZER D. SINGPURWALLA

JULY 1983

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

U. S. ARMY MATERIEL SYSTEMS ANALYSIS ACTIVITY
ABERDEEN PROVING GROUND, MARYLAND

DISPOSITION

Destroy this report when no longer needed. Do not return it to the originator.

DISCLAIMER

The findings in this report are not to be construed as an official Department of the Army position unless so specified by other official documentation.

WARNING

Information and data contained in this document are based on the input available at the time of preparation. The results may be subject to change and should not be construed as representing the DARCOM position unless so specified.

TRADE NAMES

The use of trade names in this report does not constitute an official endorsement or approval of the use of such commercial hardware or software. The report may not be cited for purposes of advertisement.

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER TR 373	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) An Empirically Developed Fourier Series Model For Describing Software Failures		5. TYPE OF REPORT & PERIOD COVERED 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Larry H. Crow Nozer D. Singpurwalla		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS US Army Materiel Systems Analysis Activity Aberdeen Proving Ground, MD 21005-5071		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS US Army Materiel Systems Analysis Activity Aberdeen Proving Ground, MD 21005		12. REPORT DATE July 1983 13. NUMBER OF PAGES 38
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office) US Army Materiel Development & Readiness Command 5001 Eisenhower Avenue Alexandria, VA 22333		15. SECURITY CLASS. (of this report) UNCLASSIFIED 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Software failures
Software development
Clustered data
Reliability
Fourier Series
Spectrogram

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

In this paper, we draw attention to the fact that the times between failures of a large software system, between any two program fixes, need not be independently and exponentially distributed, as has been often assumed in the past. We argue that, in several instances, such times between failures occur in clusters, often systematically, and present some data to substantiate this claim. We then propose an empirically developed Fourier series model, which can provide an adequate description of our data, and which under certain

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

20. circumstances can be used to predict future failures. Much of our analysis here is informal, and the key tool that we use to develop our approach is a spectrogram of the data.

ACKNOWLEDGEMENTS

The US Army Materiel Systems Analysis Activity (AMSAA) recognizes the following individual for contributing to this report:

Peer Reviewer: Mr. Joseph Wald

We acknowledge the efforts of Mr. Refik Soyer and Robin Meader of George Washington University in connection with several aspects of the work reported here.

The excellent typing of Ms. Sandra Johnson and Mrs. Patricia Andrews of AMSAA are also acknowledged.

Next page is blank.

CONTENTS

Next page is blank.

AN EMPIRICALLY DEVELOPED FOURIER SERIES MODEL FOR DESCRIBING SOFTWARE FAILURES

1. INTRODUCTION

During the development program for complex computer software, it is typical for the software to be subjected to a series of test phases to uncover existing problem areas. Because of various physical constraints on the development program, it is common for the software configuration to be held fixed during a test phase and for modifications to be incorporated as a group at the end of each phase. Upon the occurrence of a failure during test, the computer system would be re-initialized and then allowed to continue operation. The investigation reported here was undertaken for the purpose of analyzing the stochastic behavior of computer software failures between modifications of the software configuration.

Much of the published literature on software reliability modelling does not seriously address this issue. It makes the convenient assumption that in any time interval between program fixes, the times between program failures are *independently* and identically exponentially distributed. For instance, Jelinski and Moranda (1972), Shooman (1972), Littlewood and Verrall (1973), Forman and Singpurwalla (1977), and Langberg and Singpurwalla (1982), all make this assumption. Whereas the assumption may be reasonable under certain circumstances, especially those in which a fix is made every time the program fails, there do exist situations for which it may not be appropriate. To support our claim, we refer the reader to a classic paper by Lewis (1964) in which it is maintained that often hardware and software failures occur in bunches, or <u>clusters</u>.

For hardware failures, the clustering phenomenon may be attributed to imperfect repair [Brown and Proschan (1982)], or to minimal repair [Balaban and Singpurwalla (1982)]. In the case of software failures, the clustering may be caused by variations in the operating environment [cf. Gaver (1963)]. For instance, the nature of demands made on the software changes over time with a tendency for similar types of demands occurring close to each other -- this may result in a succession of failures.

In this paper we first make a case for the point of view that the independence assumption may not always be true, and that software failures between program fixes can occur as a series of clusters. This implies that a distribution, such as the exponential, cannot be used, in general, as a model for software reliability. It is noted also that in the presence of clustering, the often used hardware concept of mean time between failure (MTBF) to characterize software reliability may not be appropriate. Given this situation, we would like to suggest the Fourier series model as a possible tool for analyzing clustered software data. For a particular set of data, the Fourier series model, and its corresponding spectrogram, may be used to describe the degree of the clustering phenomenon. Our attitude regarding this model is to emphasize "data analysis," rather than "statistical inference." In addition, we remark that if the Fourier series model provides a good fit to the observed data, and if we expect the pattern to continue, then it may also be a vehicle for providing insight into future times to failure. This is illustrated by example in Section 5.

In order to achieve the above objectives we undertake the following:

1. Display several sets of real life data on the times between failures of a software system, collected under carefully controlled conditions, which clearly reveal clustering.
2. Recommend using the "spectrogram," a device routinely used in time series analysis, for determining if there is clustering, and if so, whether the clustering is systematic (periodic) or not.
3. Demonstrate how the spectrogram can also be used to empirically develop a Fourier series model which can capture the essential features of the failure behavior, and which may be used for predicting the future times to failure.

By way of a conclusion, we state that new software reliability models which are capable of incorporating the effect of clustering need to be developed. Once this is done, more formal procedures of statistical inference can be embarked upon.

2. ON CLUSTERING OF SOFTWARE FAILURES

Basically, there are two types of computer systems. The first type is one in which the operational environment does not influence the performance of the software. This means that if a request is made of the software at time t, it will give a certain response, and if the same request is made at a later time t + s, it will give the same response. The operational environment of the system does not impact on the response given by the software. The second type of computer system is one in which the operational environment does influence the performance of the software. If a request is made from this type of system at time t and the same request is made at time $t + s$, the software system may give different responses due to the influence of the operational environment between times t and $t + s$.

We call the first type of computer system deterministic state and the second type stochastic state. The state of the computer at time t is the computer memory together with the logic step of the program at time t. This completely characterizes the computer system at time t. For a deterministic state computer system, the state of the computer at the time of a request does not depend on the physical environment in which the system operates. The state of the computer for a stochastic state system does depend on the physical environment in which the system operates.

In this paper we are interested in the behavior of failures for complex stochastic state computer systems. For these systems the state of the computer at times t and $t + s$ will generally be similar for small s. In addition, the environment which generates the requests will typically be similar over this time period. Now, a computer failure is caused by the inability of the software to perform a particular request in its current state. Consequently, if a computer failure occurs at time t there would tend to be an increased chance that another failure will occur in the near term. One may conclude, therefore, that failures for stochastic state computer systems will tend to occur in clusters in an operational environment.

3. DESCRIPTION OF THE DATA ANALYZED

In the next section we analyze software failure data generated by two complex military, stochastic state, computer systems. We refer to these as System A and System B. The two data sets for System A were obtained from two copies of this system operated during the same time period under similar operational environments. The data set from System B represents software failures which occurred on one copy of this system.

Both of these systems were tested under controlled operational conditions. For these systems there are two types of software problems that may occur. The first type is a software error which degrades the operation of the system. This includes, for example, incidents in which all processing by the system is ceased and incidents in which processing appears to be continuing but the operator is unable to enter anything. It would also include incidents, due to the software, in which the system fails to carry out a task, such as transmit a message. These problems are generally obvious. The other type includes incidents of software error which yield incorrect results without inhibiting the system operation. These are more difficult to detect.

The incidents recorded and analyzed for Systems A and B were of the first type. The times of occurrence of these anomalies were observed and are given in Tables 1, 2, and 3, and shown in Figures A.1, A.2, and A.3 of the Appendix.

4. AN OUTLINE OF THE METHOD OF ANALYSIS

In a simple way, <u>clustering</u> can be defined as a grouping of similar objects. Since we are dealing with software failures, we say that software failures occur in clusters, if failures have a tendency to occur in groups. One way to describe a grouping is to observe if the times between successive failures are short for a certain number of failures and long for the remaining ones.

Clustering can be either systematic, approximately systematic, or neither, depending on the environment which induces failures. Let $T_1, T_2, \ldots, T_n$ denote the successive times of software failures between any two program fixes; note that $T_1 \leqq T_2 \leqq \cdots \leqq T_{n+1}$. By plotting $T_1, T_2, \ldots, T_{n+1}$ on a time axis, we can observe if there is any form of clustering; this is one of the most elementary tests of clustering. However, the existence of a pattern in clustering is not always easy to establish. For this, we use some of the techniques of time series analysis. A <u>time series</u> is a sequence of observations which are indexed by time. The key feature of time series analysis which distinguishes it from other statistical analysis is a recognition of the fact that the observations in a time series arrive according to some order. If we let $t_i = T_{i+1} - T_i$, $i = 1,2,\ldots,n$, then the sequence of times between software failures $\{t_i\}$ indexed by $i = 1,2,\ldots,n$ is a time series. If the t_i's are short for a group of successive failures, and long for the remaining ones, then this is an indication of clustering. The question that we need to address now pertains to whether this repetition of short and long inter-failure times is systematic or not. That is, we need to investigate if there exists an embedded *period* in the process generating the clusters. One way of answering this question is to find out if there is an underlying "cyclical trend" [see Anderson and Singpurwalla (1980) -- henceforth (AS)].

4.1 Cyclical Trends

A trend is defined as a broad movement in a time series. In many series, the trend $f(i)$, a function of the index i, repeats itself after a certain time interval called the period. When this happens, the trend is called a cyclical trend.

For our analysis of $\{t_i\}$, $i = 1,\ldots,n$, the times between software failures, we shall assume that

$$t_i = f(i) + \varepsilon_i , \qquad i = 1,\ldots,n ,$$

where ε_i is a disturbance term assumed to have mean 0 and constant variance.

To capture the cyclical pattern in $f(i)$, *if any*, it is convenient to express $f(i)$ as a linear combination of sine and cosine terms. This is known as a Fourier series representation of $f(i)$. The trigonometric functions $\sin\lambda\mu$ and $\text{cosine}\lambda\mu$ being periodic, with period $2\pi/\lambda$, are convenient for describing the cyclical behavior of $f(i)$. The reciprocal of the period is called the frequency; it denotes the number of periods in a unit interval.

In order to obtain a Fourier series representation of $f(i)$, assume that n is odd, and let $q = (n-1)/2$. Then, for $k_j = j$, $j = 1,2,\ldots,q$, we may write [see (AS)]

$$f(i) = \alpha_0 + \sum_{j=1}^{q} \left[\alpha(k_j) \cos \frac{2\pi}{n} k_j i + \beta(k_j) \sin \frac{2\pi}{n} k_j i\right]$$

where the coefficients α_0, $\alpha(k_j)$ and $\beta(k_j)$ are obtained using the principle of least squares as:

$$\alpha_0 = \frac{1}{n} \sum_{i=1}^{n} t_i ,$$

$$\alpha(k_j) = \frac{2}{n} \sum_{i=1}^{n} t_i \cos \frac{2\pi}{n} k_j i , \text{ and}$$

$$\beta(k_j) = \frac{2}{n} \sum_{i=1}^{n} t_i \sin \frac{2\pi}{n} k_j i , \qquad j = 1,2,\ldots,q .$$

A plot of $\rho^2(k_j) \stackrel{\text{def}}{=} \alpha^2(k_j) + \beta^2(k_j)$, versus the frequency k_j/n , for $j = 1,2,\ldots,q$, is called a <u>spectrogram</u> of the series $\{t_i\}$. The quantity $\rho(k_j)$ is a measure of how closely the trigonometric function with frequency k_j/n fits the observed series. Note that if a series of length n has a period ϕ , then the value $\rho(k_j)$ corresponding to $k_j = n/\phi$ will tend to be the *largest* among all other $\rho(k_j)$. Thus the spectrogram can be used to discover hidden periods in a time series by identifying the frequencies k_j/n associated with values of $\rho(k_j)$ which are visibly larger than the others. For even values of n , the procedure for obtaining the spectrogram is exactly the same as above, except that now q is $n/2$, and $\cos \frac{2\pi q}{n} i$ and $\sin \frac{2\pi q}{n} i$ simplify to $(-1)^i$ and 0, respectively; the coefficient $\alpha(q)$ simplifies to $\frac{1}{n} \sum_{i=1}^{n} (-1)^i t_i$.

In order to see if the series has a single dominant period, and if so, to specify its value, or to see if the series has multiple periods or is even aperiodic, a more detailed examination of the spectrogram is necessary. Furthermore, the spectrogram can also be used to obtain a parsimonious model which adequately describes the time series. These and other matters are discussed in the next two sections.

4.2 Interpreting the Spectrogram

Suppose that the spectrogram of the time series $\{t_i\}$, $i = 1,\ldots,n$, reveals relatively large values of $\rho^2(k_j)$ at $k_j = i$, for $i \in I$, and $I \subset \{1,2,\ldots,q\}$; "$\subset$" denotes a subset. Let ℓ be the largest element of the set I , so that n/ℓ is the smallest among the n/i values. Note that if n is even, n/ℓ can be no smaller than 2. Now, if the n/i values are multiples of n/ℓ , then we are motivated to conclude that the series is *periodic* with a *minimum period* n/ℓ . If some of the n/i values are multiples of n/ℓ , and the remaining n/i values are multiples of another constant, then we are inclined to state that the series has *multiple periods*. In practice, of course, the values n/i will rarely be exact multiples of n/ℓ , but may be approximately so; in such cases, our specification of the minimum period n/ℓ is an approximation.

In the case of software failure data, the identification of a minimum period n/ℓ implies that there is a clustering of the failures, and that the clusters occur systematically after every n/ℓ observations. If the n/i values are approximate multiples of n/ℓ , then we say that the clusters tend to occur systematically by repeating themselves after about n/ℓ observations. If no multiplicative pattern can be discerned between n/ℓ and the other n/i values, then the clustering process is not systematic, and our parsimonious model for describing software failures (see Section 4.3) cannot be used to predict future failures. It is only useful as a descriptive tool for explaining the observed failures.

In a practical sense, the times between failures which are n/ℓ observations apart represent a measure of the time to clustering. Hence, in practice, the average of these time intervals may be indicative of a "mean time to clustering" which is a useful parameter describing the failure process. As a final comment, if the set I is indeed equal to the set $\{1,2,...,q\}$, that is, if $\rho^2(k_j)$ is equally large at all the q frequencies k_j/n, we conclude that there is *no sign of clustering* in the data, and if the $\{t_i\}$'s are *uncorrelated,* then the assumption of independence mentioned before may be appropriate.

4.3 A Parsimonious Model for Software Failures

We have stated before that the spectrogram can also be used for developing a parsimonious model which describes the underlying time series. To see how, we first remark that our Fourier series representation of the trend $f(i)$ consists of n terms, with coefficients $\alpha(k_j)$, $\beta(k_j)$, and α_0, $k_j = 1,...,q$; $q = \frac{n-1}{2}$, if n is odd, and $q = \frac{n}{2}$ if n is even. It is possible, and often very likely, that all the n terms mentioned above may not be necessary. From our description of the spectrogram, it should be clear which of the n terms are dominant and which are not. Clearly, those values of $\alpha(k_j)$ and $\beta(k_j)$ for which $\rho^2(k_j)$ is large play a dominant role in explaining our data, and these are the ones that should be used in the Fourier series model; the others can be eliminated. In the notation of Section 3.2, we have identified $k_j = j$, $j \in I$, as being such that $\rho(k_j)$ is large. Thus, our resulting parsimonious model for the times between software failures would be

$$f(i) = \alpha_0 + \sum_{k_j=i,i,\varepsilon I} \left[\alpha(k_j) \cos \frac{2\pi}{n} k_j i + \beta(k_j) \sin \frac{2\pi}{n} k_j i\right] .$$

In practice, we would hope that the number of elements of I is much smaller than $\frac{n-1}{2}$ (or $\frac{n}{2}$). Furthermore, a model of the form given above is easier to justify in the context of software failures, if we have identified a minimum period n/ℓ, since now we are saying that the clustering is systematic and so it can be reasonably well predicted.

An important part of the analysis is to plot the estimated Fourier series $f(i)$ against the observed data t_i to determine visually how well the model fits the data. In addition, statistical tests can often be employed to test hypotheses regarding the terms to be included in the fitted model. For example, under certain general assumptions a Chi-Squared statistic can be used to test the null hypothesis that a cyclical term with minimum period η/k_j exists, for j specified. Also, an F-statistic can be used to test whether any cyclical trends at all are evident in the data. For a discussion of these tests see Anderson (1971) p. 101 and (AS) p. 90.

5. APPLICATION TO SOFTWARE FAILURE DATA

The methodology described in Section 3 will now be applied to the software failure data discussed in Section 3. Recall that there are three sets of data, the first and second sets pertaining to the same system run under two different environments. The data are obtained in terms of times to software failures $T_1 \leqq T_2 \leqq \cdots \leqq T_{n+1}$. For our analysis, we consider the times between failures $t_i = T_{i+1} - T_i$, and consider the time series generated by the sequence $\{t_i\}$, $i = 1,\ldots,n$. A plot of the three time series under consideration is given by the faint lines of Figures 5.1, 5.2, and 5.5.

5.1 Analysis of the First Set of Data

The first set of data consists of 77 software failures, and so our time series will consist of 76 times between failures. An examination of Figure 5.1 reveals several regularly occurring peaked values in the series; this suggests some periodicities (clusters) in the data. This is also suggested by Figure A.1. In Figure 5.3, we show a spectrogram of this series for the range of frequencies $\frac{1}{76}, \frac{2}{76}, \ldots, \frac{38}{76} = .5$. If these data had a period of 3, then this period and its harmonics at 6, 9, 12, 15, ..., would imply that the values of $\rho^2(k_j)$ at, or in the vicinity of the frequencies $\frac{25}{76}, \frac{13}{76}, \frac{8}{76}, \frac{6}{76}, \frac{5}{76}, \frac{4}{76}$, and $\frac{2}{76}$, would tend to be large. In Figure 5.3 we observe that large values of $\rho^2(k_j)$ occur at the frequencies $\frac{1}{76}, \frac{2}{76}, \frac{6}{76}, \frac{8}{76}, \frac{13}{76}, \frac{21}{76}, \frac{22}{76}, \frac{26}{76}, \frac{27}{76}, \frac{28}{76}$, and $\frac{35}{76}$. Of these frequencies, $\frac{2}{76}, \frac{6}{76}, \frac{8}{76}, \frac{13}{76}, \frac{22}{76}, \frac{26}{76}, \frac{27}{76}$, and $\frac{28}{76}$, would be close in terms of approximating a period of 3. These frequencies are flagged by a diamond on Figure 5.3. If these flagged frequencies are the only ones that are used in a Fourier series model for the trend $f(i)$, then our model for describing the times between software failures turns out to be:

$$\begin{aligned}
\hat{f}(i) = 4.3954 &+ 1.7969 \cos\left(2\pi \tfrac{2}{76} i\right) - 0.0242 \sin\left(2\pi \tfrac{2}{76} i\right) \\
&+ 1.3201 \cos\left(2\pi \tfrac{6}{76} i\right) - 0.8049 \sin\left(2\pi \tfrac{6}{76} i\right) \\
&+ 0.6163 \cos\left(2\pi \tfrac{8}{76} i\right) - 1.7054 \sin\left(2\pi \tfrac{8}{76} i\right) \\
&- 0.8120 \cos\left(2\pi \tfrac{13}{76} i\right) - 1.2849 \sin\left(2\pi \tfrac{13}{76} i\right) \\
&+ 0.7595 \cos\left(2\pi \tfrac{22}{76} i\right) + 1.4534 \sin\left(2\pi \tfrac{22}{76} i\right) \\
&- 1.0774 \cos\left(2\pi \tfrac{26}{76} i\right) + 1.1533 \sin\left(2\pi \tfrac{26}{76} i\right) \\
&+ 1.1037 \cos\left(2\pi \tfrac{27}{76} i\right) - 1.6733 \sin\left(2\pi \tfrac{27}{76} i\right) \\
&- 1.5472 \cos\left(2\pi \tfrac{28}{76} i\right) - 0.4100 \sin\left(2\pi \tfrac{28}{76} i\right) \\
&+ 0.0386(-1)^i, \qquad \text{for } i = 1,2,\ldots,76 .
\end{aligned}$$

A plot of $\hat{f}(i)$ versus i, for $i = 1,2,\ldots,76$, is shown by the dark lines of Figure 5.1. This plot indicates that the above model provides an adequate description of the failure data.

Based on this informal analysis, we are tempted to claim that the failures in Data Set 1 occur in clusters, and that the clustering process is approximately systematic with a period of 3. Furthermore, the frequencies corresponding to the period 3 and its harmonics give us a Fourier series model which provides us with a reasonable description of the data. It is important to note that the presence of clustering means that the exponential distribution model and a corresponding MTBF is not appropriate for describing the software reliability for this system. Additionally, the period of 3 from the fitted model is indicative of the degree of clustering of the failures.

Because of the possible systematic nature of clustering, the above model could be used to give us some insight about future failures. The function $\hat{f}(i)$ is an estimate of the mean value function $f(i)$ based on the observed data. If we assume that the estimated Fourier series pattern continues past these observations, then $\hat{f}(i)$, $i = 77, \ldots,$ is a projection of the mean value function for future observations. For example, the estimated mean value for the next failure is $\hat{f}(77) = 9.8$. Also, a minimum period of 3 implies that it is likely for clustering to appear around the 78th or 79th failures. From the model this is qualified further by the projected mean values of $\hat{f}(78) = 3.8$ and $\hat{f}(79) = 1.8$, which are relatively small.

5.2 Analysis of the Second Set of Data

The second set of data consists of 67 software failures, resulting in 66 observations for a time series of times between failures. A plot of this series is shown by the faint lines of Figure 5.2. The spectrogram of these data is given in Figure 5.4. An inspection of the spectrogram indicates that these data may have multiple periods of sizes 2 and 3. The frequencies corresponding to a period of 2 and its harmonics are indicated by the *squares* and the *diamonds* in Figure 5.4. If these frequencies are used in a Fourier series model for the trend $f(i)$, then our model for describing the times between software failures turns out to be

$$\begin{aligned}\hat{f}(i) = 3.7121 &+ 0.1627 \cos(2\pi \tfrac{2}{66} i) - 1.6437 \sin(2\pi \tfrac{2}{66} i)\\ &- 1.2711 \cos(2\pi \tfrac{6}{66} i) + 0.3240 \sin(2\pi \tfrac{6}{66} i)\\ &+ 0.3265 \cos(2\pi \tfrac{16}{66} i) + 0.9928 \sin(2\pi \tfrac{16}{66} i)\\ &+ 1.0276 \cos(2\pi \tfrac{21}{66} i) - 0.8693 \sin(2\pi \tfrac{21}{66} i)\end{aligned}$$

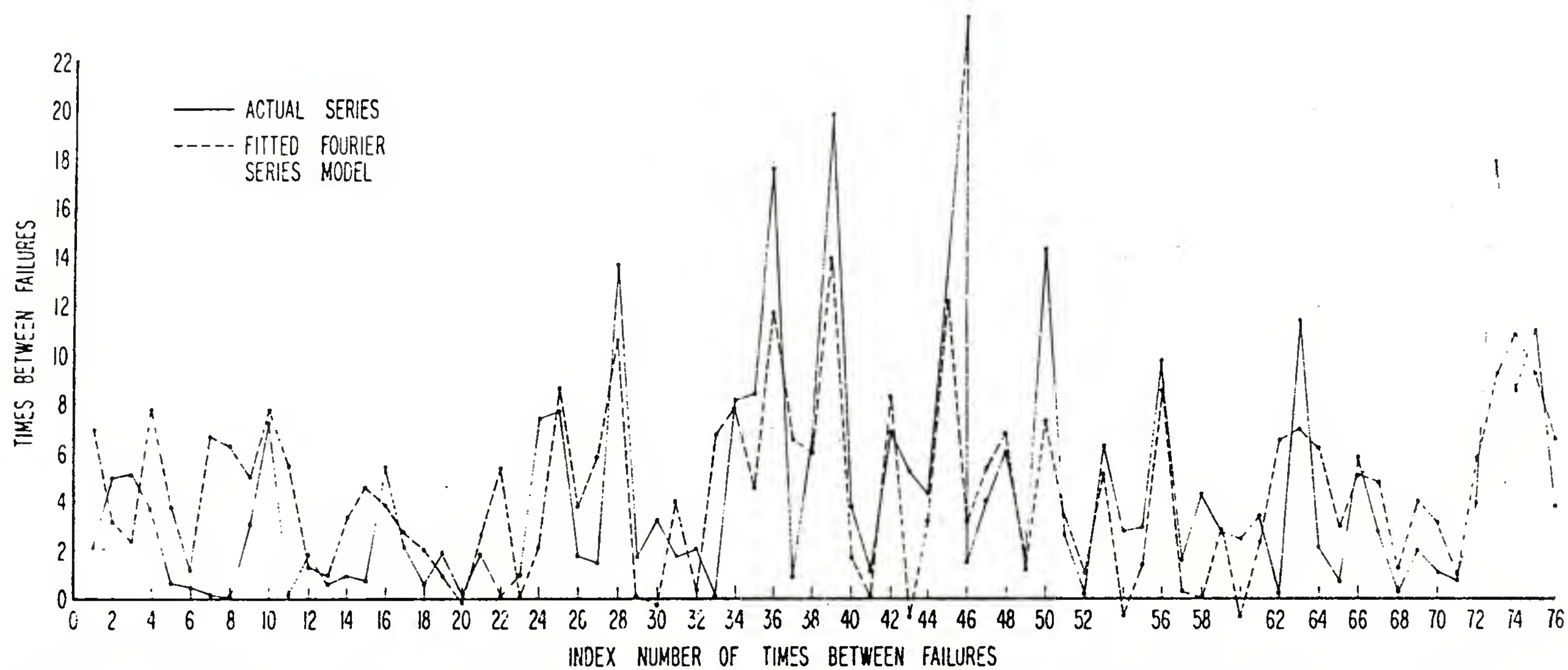

Figure 5.1. A Time Sequence Plot Showing the Times Between Failures for System A, Data Set 1, and Their Fitted Values Using a Fourier Series Model.

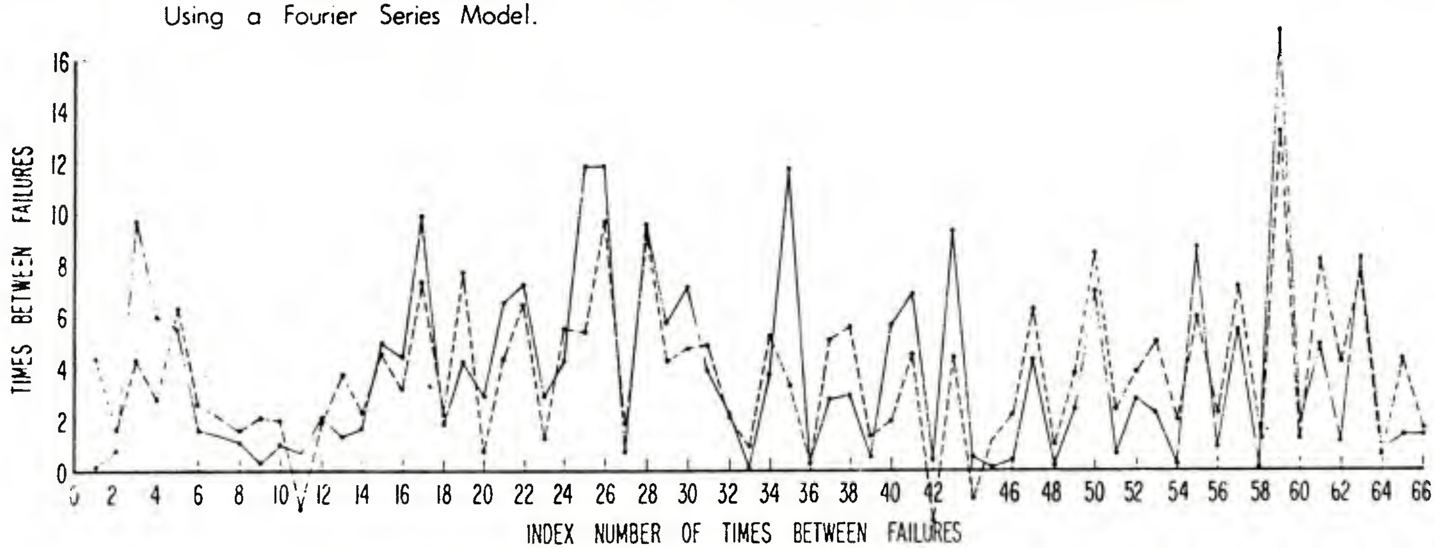

Figure 5.2. A Time Sequence Plot Showing the Times Between Failures for System A, Data Set 2, and Their Fitted Values Using a Fourier Series Model.

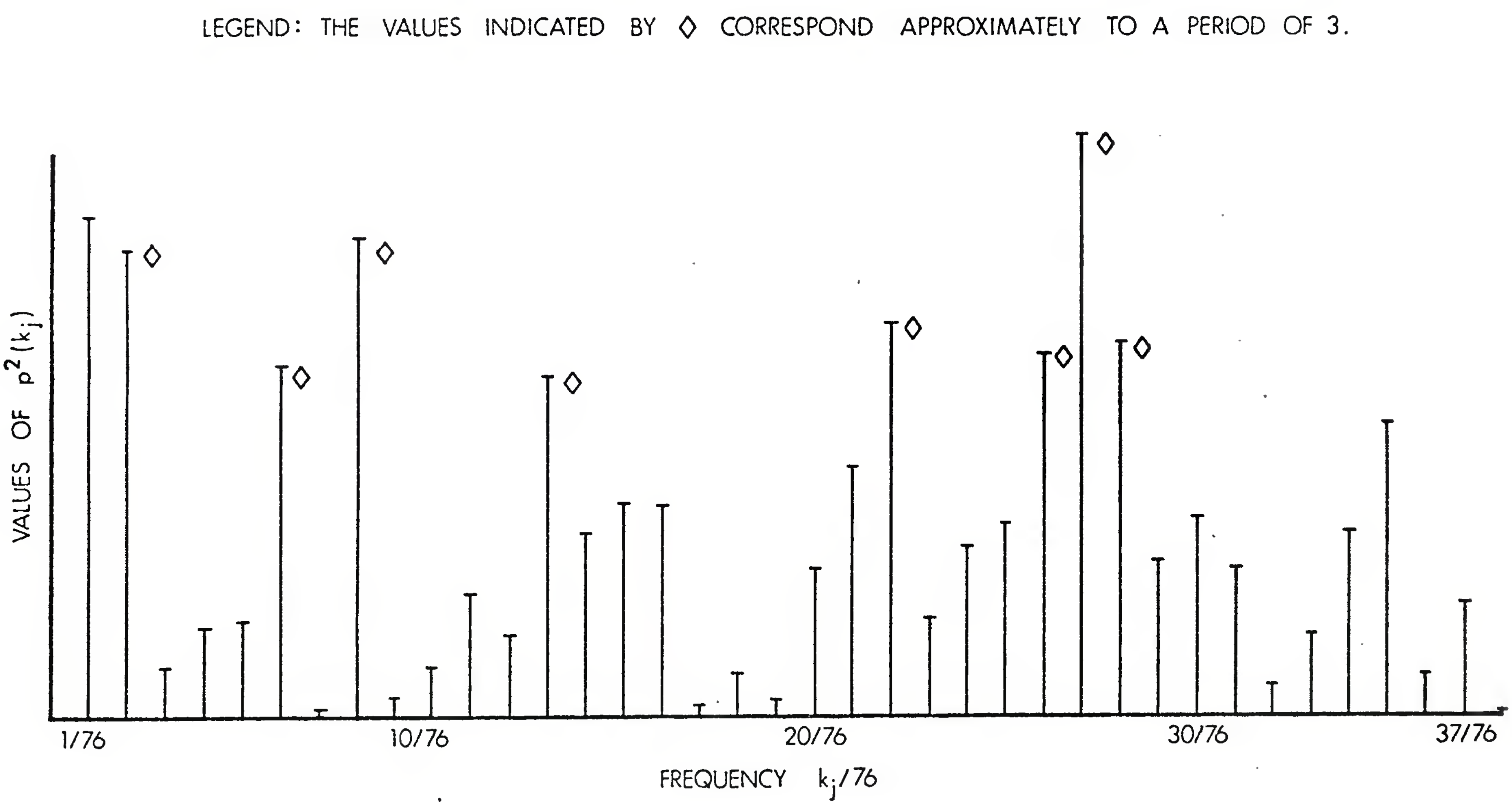

Figure 5.3. A Spectrogram of the Times Between Failures for System A, Data Set 1.

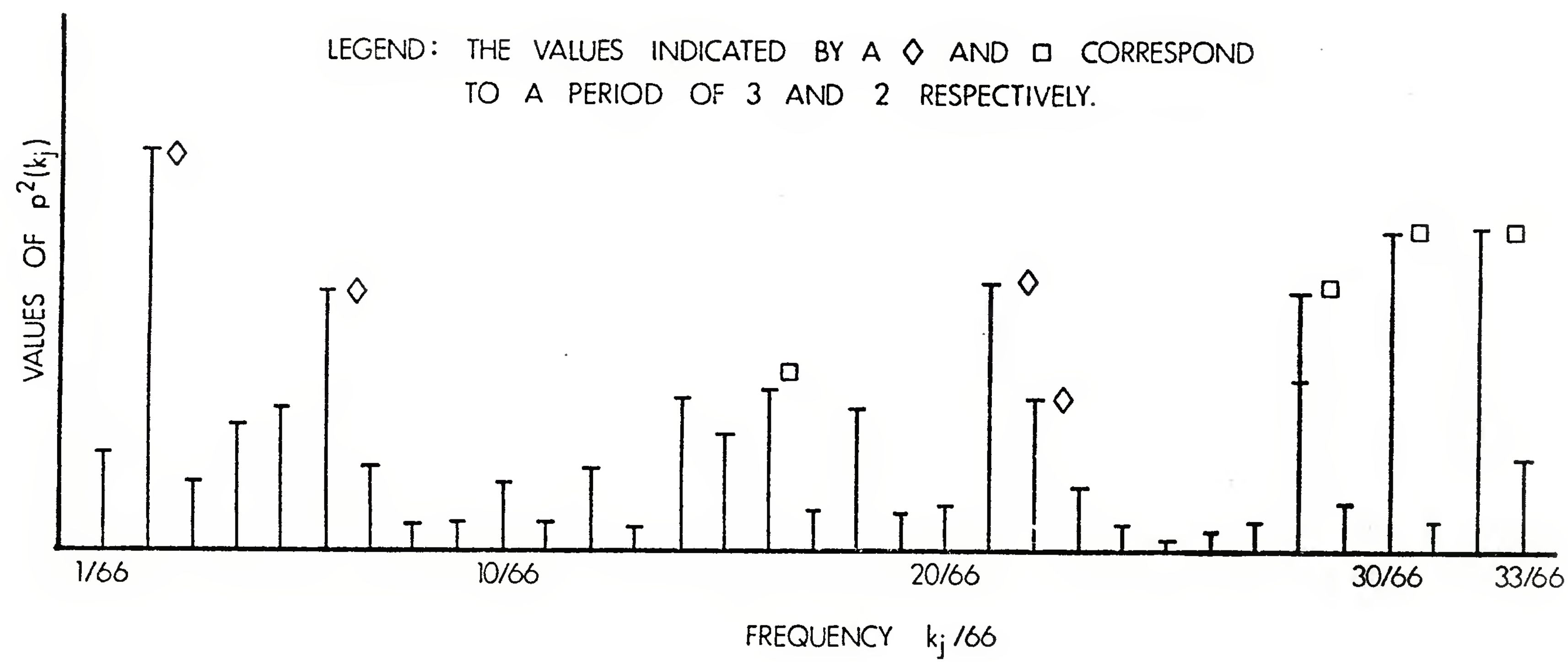

Figure 5.4. A Spectrogram of the Times Between Failures for System A, Data Set 2.

$$- 1.0121 \cos(2\pi \frac{22}{66} i) + 0.0315 \sin(2\pi \frac{22}{66} i)$$

$$+ 0.9574 \cos(2\pi \frac{28}{66} i) + 0.9190 \sin(2\pi \frac{28}{66} i)$$

$$- 0.3434 \cos(2\pi \frac{30}{66} i) - 1.4211 \sin(2\pi \frac{30}{66} i)$$

$$- 1.2577 \cos(2\pi \frac{32}{66} i) - 0.7710 \sin(2\pi \frac{32}{66} i)$$

$$- 0.7788(-1)^i , \qquad i = 1,2,\ldots,66 .$$

A plot of $\hat{f}(i)$ versus i , $i = 1,2,\ldots,66$, is shown by the dark lines of Figure 5.2. This plot indicates that the above model provides an adequate description of the failure data. There are, however, three instances in which the fitted model yields negative times between failures. This possibility is the nature of a Fourier series model and has to be judged in the light of the otherwise good description of the data that such a model provides. Negative values given by the model are generally indicative of clustering and the corresponding short times between failures. A future time between failure which is predicted by the model to be negative could, therefore, be interpreted as one which is expected to be positive but relatively short. From a practical point of view, a likely range on the actual magnitude of this time between failure may be indicated from previous times between failures within clusters.

Our conclusions at the end of Section 5.1 apply here also.

5.3 Analysis of the Third Set of Data

The third set of data consists of 57 software failures, resulting in 56 observations for our time series. A plot of the time series is shown by the faint lines of Figure 5.5, and its spectrogram in Figure 5.6. An inspection of the spectrogram suggests a period of 2 and its harmonics. The frequencies corresponding to a period 2 are flagged by the *diamonds* in Figure 5.6. A Fourier series model corresponding to these frequencies is given by:

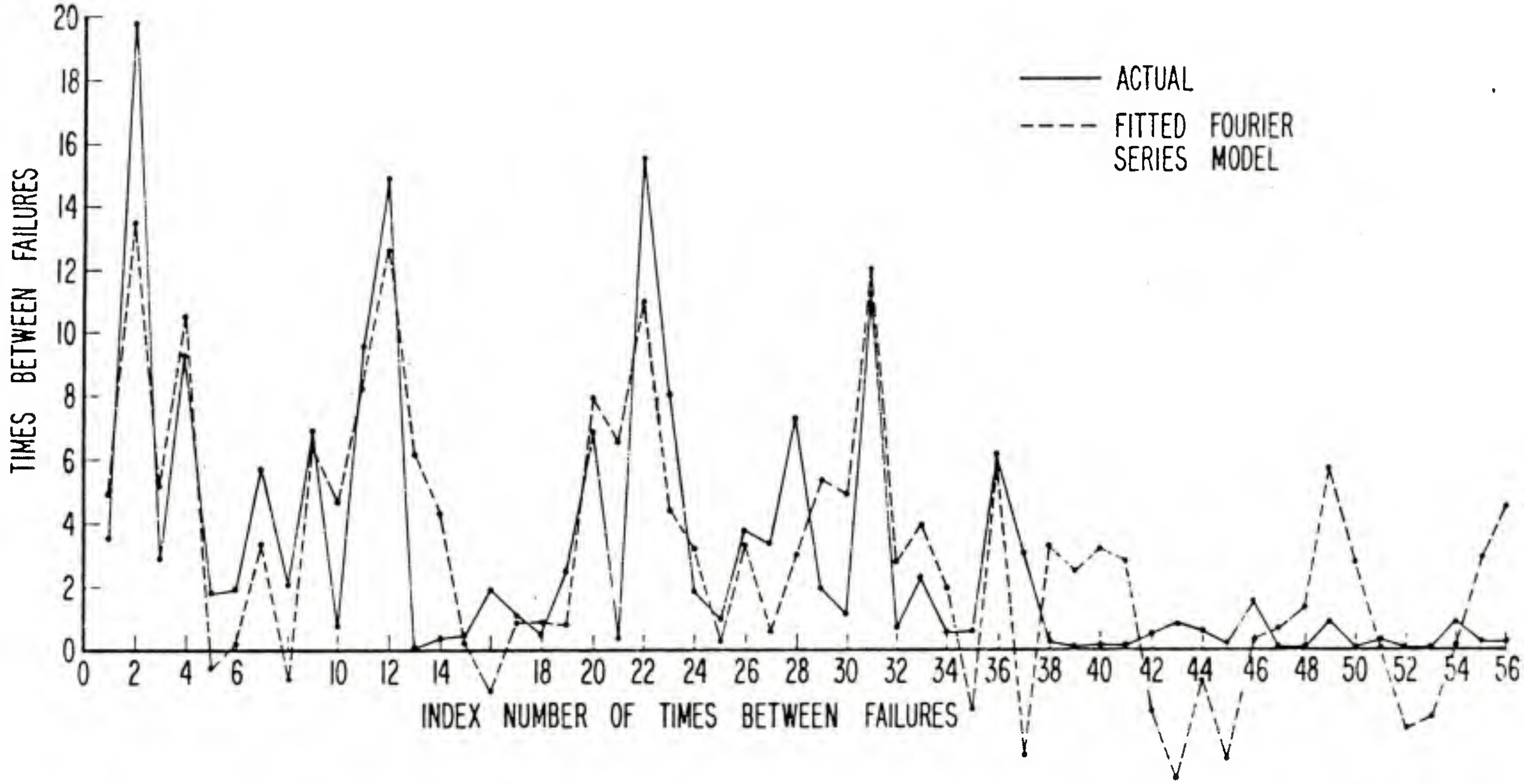

Figure 5.5. A Time Sequence Plot Showing the Times Between Failures for System B, and Their Fitted Values Using a Fourier Series Model.

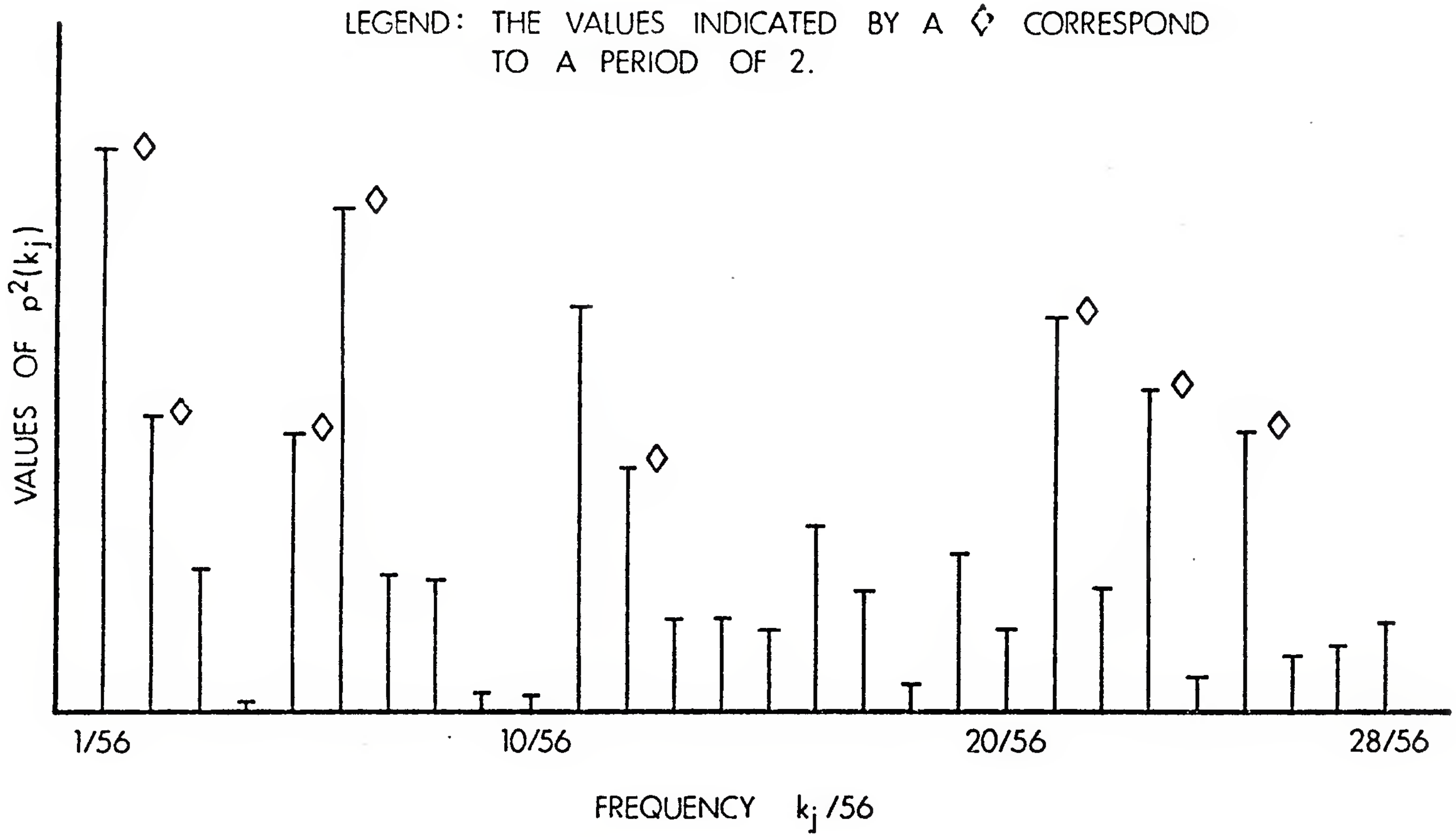

Figure 5.6. A Spectrogram of the Times Between Failures for System B.

$$
\begin{aligned}
f(i) \;=\; & 3.0446 - 0.3344\cos\left(2\pi\tfrac{1}{56}i\right) + 2.0282\sin\left(2\pi\tfrac{1}{56}i\right) \\
& + 1.1113\cos\left(2\pi\tfrac{2}{56}i\right) + 0.9867\sin\left(2\pi\tfrac{2}{56}i\right) \\
& + 1.3400\cos\left(2\pi\tfrac{5}{56}i\right) + 0.5421\sin\left(2\pi\tfrac{5}{56}i\right) \\
& + 0.0966\cos\left(2\pi\tfrac{6}{56}i\right) + 3.7896\sin\left(2\pi\tfrac{6}{56}i\right) \\
& - 0.9992\cos\left(2\pi\tfrac{12}{56}i\right) - 0.9037\sin\left(2\pi\tfrac{12}{56}i\right) \\
& - 1.7157\cos\left(2\pi\tfrac{21}{56}i\right) - 0.0807\sin\left(2\pi\tfrac{21}{56}i\right) \\
& + 0.4478\cos\left(2\pi\tfrac{23}{56}i\right) - 1.4830\sin\left(2\pi\tfrac{23}{56}i\right) \\
& + 1.1162\cos\left(2\pi\tfrac{25}{56}i\right) - 0.9358\sin\left(2\pi\tfrac{25}{56}i\right) \\
& + 0.5518(-1)^{i}, \qquad i = 1,2,\ldots,56 .
\end{aligned}
$$

A plot of $\hat{f}(i)$ versus i, for $i = 1,2,\ldots,56$, is shown by the dark lines of Figure 5.6. The plot suggests that the above model provides an adequate description of much of the data. The model fails to capture the latter part of the data, and this may be due to the excessive clustering towards the end. Such behavior of the data may be responsible for destroying the systematic clustering with period 2 which the model attempts to incorporate. In light of such behavior, using this model for predictive purposes is <u>not recommended</u>, unless of course there is reason to believe that the excessive clustering at the end is temporary, and can be eliminated.

6. CONCLUSIONS

Our analysis of the three sets of data has shown that there exists a cyclical trend in the time between software failures. In all three series, the data have been successfully described by the estimated cyclical trend. This result is an indication of the existence of systematic clustering in software failures, and thus our claim that the assumption of independence and identical exponential distribution for the times between failures may not always hold.

A model for describing software failures which attempts to incorporate the effects observed in the data presented here is needed, and the authors are currently working on this development.

Next page is blank.

REFERENCES

1. Anderson, T. W. (1971). The Statistical Analysis of Time Series. John Wiley and Sons, Inc., New York.

2. Anderson, T. W. and N. D. Singpurwalla (1980). Methods and applications of time series analysis: Part I: Regression, trends, smoothing and differencing. Technical Report No. 42, Department of Statistics, Stanford University, Stanford, California.

3. Forman, E. H. and N. D. Singpurwalla (1977). An empirical stopping rule for debugging and testing computer software. *Journal of the American statistical Association*, Vol. 72, pp. 750-757.

4. Gaver, D. P. (1963). Random hazard in reliability problems. *Technometrics*, Vol. 5, No. 2, pp. 211-226.

5. Jelinski, Z. and P. B. Moranda (1972). Software reliability research, in *Statistical Computer Performance Evaluation* (W. Freiberger, ed.), pp. 485-502. Academic Press, New York.

6. Langberg, N. and N. D. Singpurwalla (1982). Unification of some software reliability models via the Bayesian approach. To appear.

7. Littlewood, B. and J. L. Verrall (1973). A Bayesian reliability growth model for computer software. *Record IEEE Symposium on Computer Software Reliability*, pp. 70-77. IEEE, New York.

Next page is blank.

APPENDIX

Table A.1. Times of Software Errors

n = 77, Total Test Time T = 343.65

.50	28.18	44.82	87.05	168.95	235.35	281.75
2.53	30.05	45.87	95.27	192.75	245.15	284.55
7.50	30.67	53.33	103.93	194.05	246.75	284.95
12.63	31.70	61.15	121.75	198.05	251.15	287.05
16.25	32.45	62.88	122.75	204.85	253.95	288.15
16.92	38.00	64.40	129.15	206.35	256.65	288.85
17.40	40.23	78.17	148.95	220.65	260.35	292.95
17.63	40.77	79.93	152.75	223.25	260.55	311.05
17.67	42.73	83.22	153.75	223.45	273.05	319.55
20.80	42.90	84.97	160.65	229.75	275.15	330.75
28.13	44.78	87.03	165.85	232.45	275.85	334.55

Table A.2. Times of Software Errors

n = 67, Total Test Time T = 331.0

4.4	33.2	80.8	194.1	239.6	268.4
4.6	35.4	83.7	197.9	239.7	269.3
5.4	36.8	88.0	209.8	240.1	274.8
15.2	38.5	118.0	210.2	244.4	274.9
21.2	43.5	164.0	213.1	244.5	292.1
26.7	47.9	164.9	216.1	246.9	293.9
28.4	57.9	174.6	216.7	253.9	298.7
29.9	59.7	180.4	222.4	254.5	299.8
31.1	64.0	187.6	229.4	257.3	308.1
31.4	66.9	191.6	229.8	259.5	308.8
32.5	73.5	193.9	239.1	259.7	310.2
					311.6

Table A.3. Times of Software Errors

n = 57, Total Test Time T = 202.0

25.7	72.8	107.9	144.9	175.5	189.0	192.9
29.2	79.7	109.0	145.9	177.9	189.1	193.9
49.0	80.4	109.5	149.7	178.5	189.6	194.0
51.9	90.0	112.0	153.1	179.1	190.4	194.4
61.2	105.0	118.9	160.4	185.3	191.0	194.5
63.0	105.0	119.3	162.4	188.4	191.2	194.6
64.9	105.4	134.9	163.6	188.7	192.7	195.6
70.7	105.9	143.0	174.8	188.8	192.7	195.9
						196.2

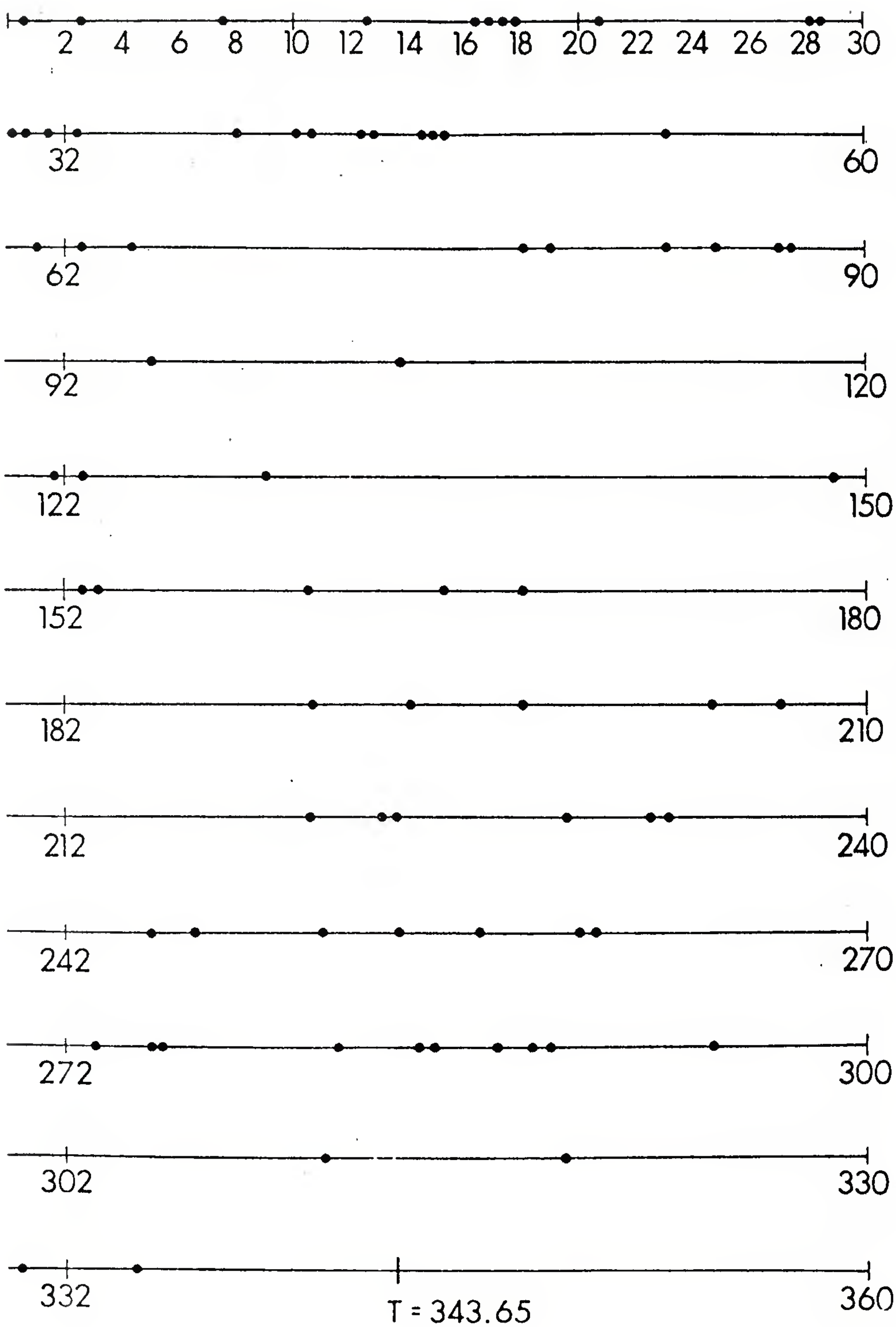

Figure A.1. A Plot of the Failure Times for System A, Data Set 1.

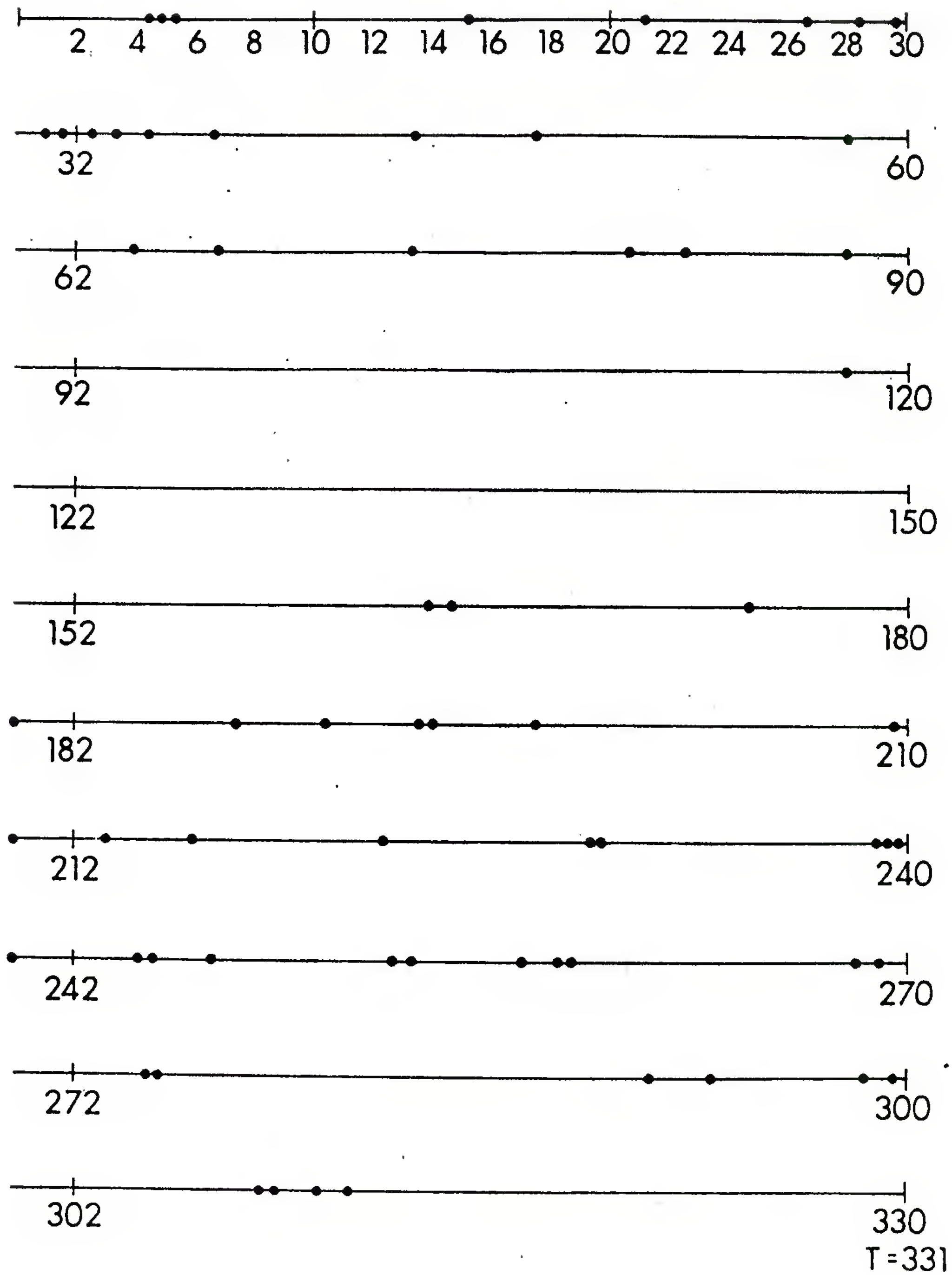

Figure A.2. A Plot of the Failure Times for System A, Data Set 2.

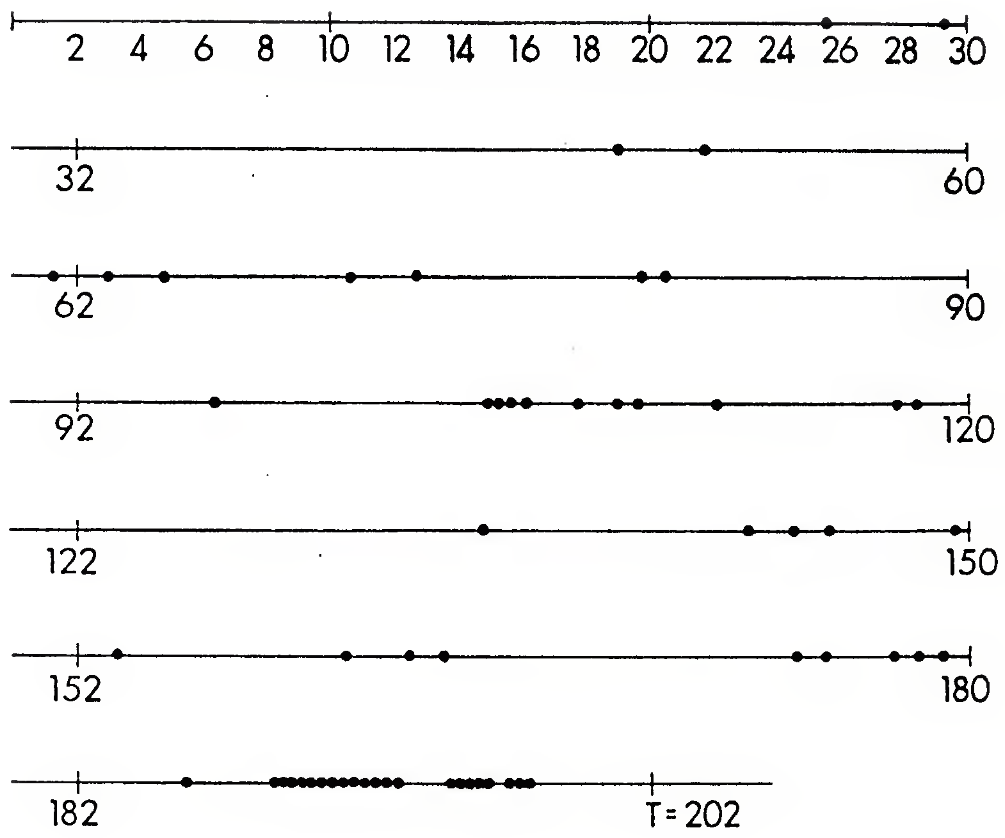

Figure A.3. A Plot of the Failure Times for System B.

DISTRIBUTION LIST

No. of Copies	Organization
12	Commander Defense Technical Information Center ATTN: DDC-TC Cameron Station Alexandria, VA 22314
1 ea.	Commander US Army Materiel Development & Readiness Command ATTN: DRCCP DRCDM-S DRCDE-F DRCRE DRCDE-A DRCQA 5001 Eisenhower Avenue Alexandria, VA 22333
2	Commander US Army Armament Research & Development Command ATTN: DRDAR-SEA Technical Library Dover, NJ 07801
1	Commander Rock Island Arsenal ATTN: Tech Lib Rock Island, IL 61299
1	Commander USAERADCOM ATTN: DRDEL-CM (Mr. W. Pepper) DELHD-SAB (old HDL) 2800 Powder Mill Road Adelphi, MD 20783
1	Commander US Army Test & Evaluation Command ATTN: STEDP-MT-L Dugway Proving Ground, UT 84022
1	Commander US Army Aviation R&D Command ATTN: DRDAV-BC 4300 Goodfellow Blvd St. Louis, MO 63120
1	Commander US Army Electronics R&D Command ATTN: DRDEL-SA Fort Monmouth, NJ 07703
1	Commander US Army Electronics R&D Command ATTN: DRDEL-AP-OA 2800 Powder Mill Road Adelphi, MD 20783
2	Director US Army TRADOC Systems Analysis Activity ATTN: ATAA-SL ATAA-T White Sands Missile Range NM 88002
1	Commander US Army Missile Command ATTN: DRSMI-DS Redstone Arsenal, AL 35898
1	Commander US Army Troop Support & Aviation Materiel Readiness Command ATTN: DRSTS-BA 4300 Goodfellow Blvd. St. Louis, MO 63120
1	Commander US Army Tank-Automotive Command ATTN: DRSTA-TSL DRSTA-V Warren, MI 48090
1	Commander US Army Mobility Equipment R&D Command ATTN: DRDME-O Fort Belvoir, VA 22060

DISTRIBUTION LIST (continued)

No. of Copies	Organization
1	Commander US Army Natick R&D Command ATTN: DRDNA-O Natick, MA 01760
1	Commander US Army Concepts Analysis Agency 8120 Woodmont Avenue Bethesda, MD 20014
1	Pentagon Library ATTN: AN-AL-RS (Army Studies) Pentagon, Room 1A518 Wash, DC 20310
3	Chief Defense Logistics Studies Information Exchange US Army Logistics Management Center ATTN: DRXMC-D (2 cys) DRXMC-ACM (Mr. Fowler) Fort Lee, VA 23801
1	Reliability Analysis Center ATTN: Mr. I. L. Krulac Griffis AFB, NY 13441

Aberdeen Proving Ground

No. of Copies	Organization
2	Cdr, USATECOM ATTN: DRSTE DRSTE-CS-A Bldg 314
	Dir, BRL, Bldg 328
	Dir, BRL ATTN: DRDAR-TSB-S (STINFO Branch) Bldg 305
	Dir, HEL, Bldg 520 ATTN: DRXHE-FSD Bldg 520

Next page is blank.